Gradient Boosting

The Art of Learning from Mistakes

Ahmed BADI

Mathematics & Machine Learning Enthusiast

ahmedbadi905@gmail.com
linkedin.com/in/badi-ahmed

December 15, 2025

Overview

Introduction

The Intuition: Learning Golf (Gradient Boosting)

- **Shot 1:** You miss 100m to the left.
- **Action:** You don't restart. You aim to fix the 100m gap.
- **Shot 2:** You correct, but are 20m short.
- **Shot 3:** You tap the ball 20m to reach the hole.

Gradient Boosting Philosophy

An iterative process where each new model learns to correct the errors (residuals) of the previous model, gradually approaching the target.

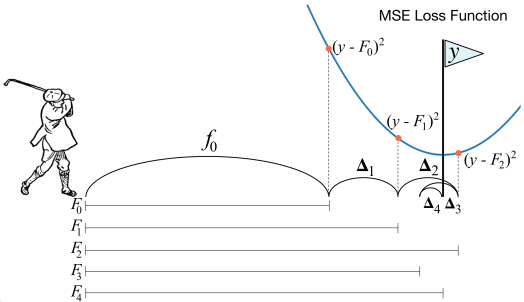
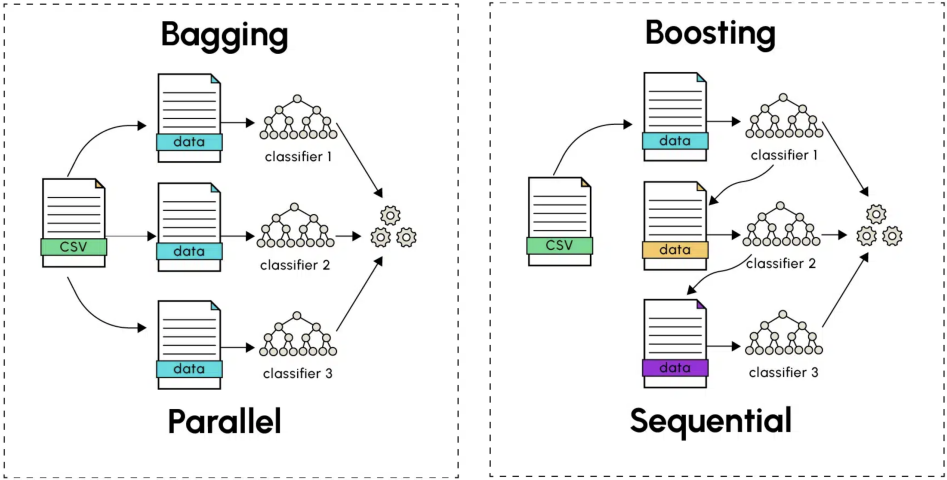


Illustration of iterative corrections: each shot fixes part of the previous error, similar to gradient boosting.

Bagging vs. Boosting



Additive Modeling

The "Team of Students" Analogy

- **Student A:** Takes the test. Scores 60%. Fails hard questions.
- **Student B:** Studies *only* the questions A failed. Fixes some errors.
- **Student C:** Studies the remaining errors.

Final Prediction = Sum of all students (Weighted).

$$F_M(x) = \sum_{m=1}^{M} \nu \cdot h_m(x)$$

Where ν is the **Learning Rate** (prevents overfitting).

Mathematics: Gradients & Residuals

Why "Gradient" Boosting?

We use Gradient Descent in Function Space.

The Connection

- **Goal:** Minimize Loss $L(y, F(x)) = \frac{1}{2}(y - F(x))^2$.
- **Gradient:** $\frac{\partial L}{\partial F(x)} = -(y - F(x))$.
- **Negative Gradient:** $y - F(x)$.

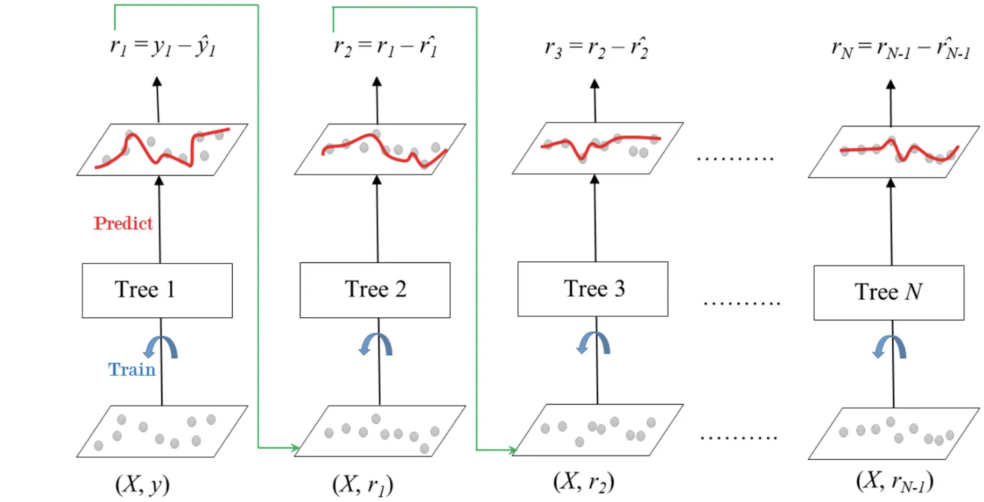
Key Insight

The **Residual** $(y - F(x))$ is actually the **Negative Gradient**.
Training on residuals $\equiv$ Training to minimize Loss via Gradient Descent.

The Algorithm Steps

1. **Initialize:** $F_0(x) = \text{mean}(y)$.
2. **For** $m = 1$ **to** M**:**
   - Calculate Residuals: $r_{im} = y_i - F_{m-1}(x_i)$.
   - Train Weak Learner (Tree) $h_m(x)$ to predict r_{im}.
   - Update: $F_m(x) = F_{m-1}(x) + \nu \cdot h_m(x)$.

Visualizing the Process



Loss Functions & Regularization

Flexible Loss Functions

GBM can optimize almost any differentiable objective.

Task	**Loss Function**	**Gradient (Target)**
Regression (Standard)	Squared Error	$y - F(x)$
Regression (Robust)	Absolute Error	$\text{sign}(y - F(x))$
Classification	Log Loss	$y - p$ (Probability)

Preventing Overfitting

GBM is "greedy". It will memorize data if allowed.

1. **Shrinkage (ν)**
   - Learning Rate (e.g., 0.01 or 0.1).
   - Slows down learning.
   - Requires more trees, but generalizes better.
2. **Tree Constraints**
   - Shallow trees (Depth 3-8).
   - Min samples per leaf.
3. **Stochastic Boosting**
   - Row subsampling.
   - Column (feature) subsampling.

The Big Three: XGBoost, LightGBM, CatBoost

The Evolution

XGBoost (2014)

- 2nd Order Gradients.
- Parallel processing.
- Handles sparsity.

LightGBM (2017)

- Histogram-based.
- Leaf-wise growth.
- Extremely fast.

CatBoost (2018)

- Native Categorical support.
- Symmetric Trees.
- Robust to default params.

Visual Results

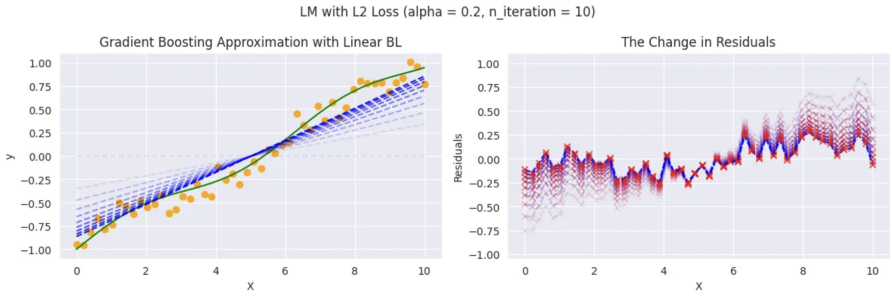


Figure: Convergence of residuals as trees are added.

Conclusion

Advantages vs Limitations

Pros

- State-of-the-art on Tabular Data.
- Flexible (Loss functions).
- Feature Importance.

Cons

- Sensitive to noise (Outliers).
- Harder to tune than Random Forest.
- Sequential training (Slower).

Thank You!

"We learn from failure, not from success."

Questions?